UNIVERSITÄT HEIDELBERG

SPADIC
Self triggered Pulse Amplification and Digitization asIC

SPADIC 1.0

Tim Armbruster
tim.armbruster@ziti.uni-heidelberg.de

SuS Meeting

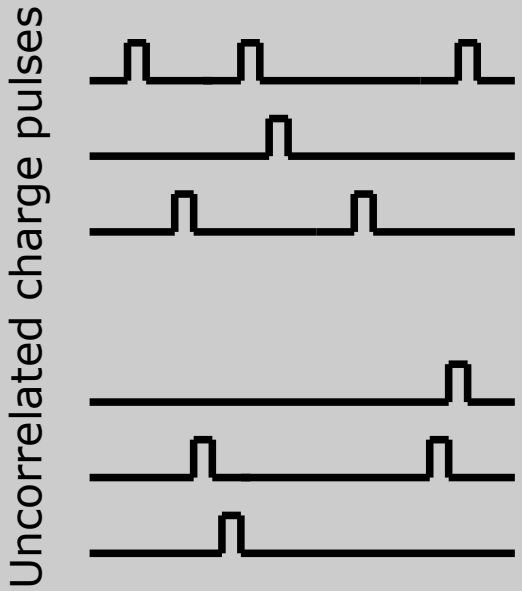
November 2011

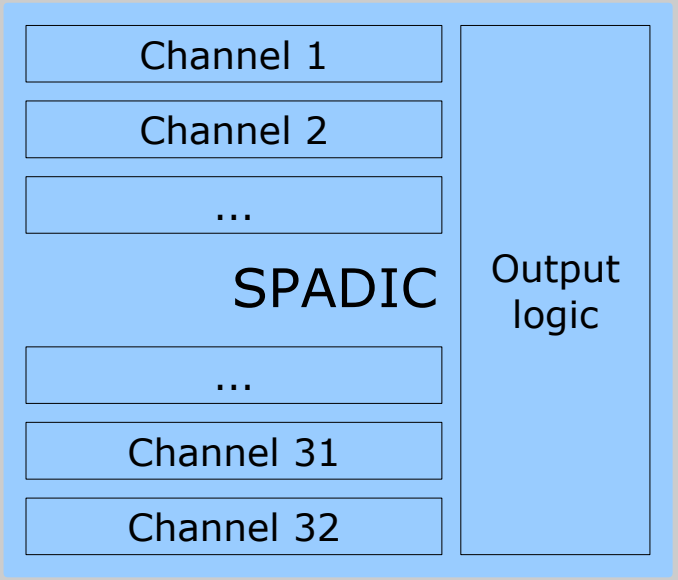
Visit http://www.spadic.uni-hd.de

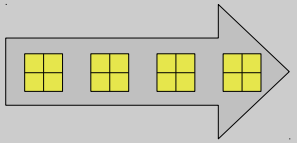
1. SPADIC Architecture

Introduction to SPADIC 1.0

SPADIC: **S**elf-triggered **P**ulse **A**mplification and **D**igitization as**IC**







Output Packages @ 1Gbps (serial)

Abstract Data Flow Concept

Charge Pulse Amplification and Shaping →
Continuous Digitization →
Continuous Filtering →
Digital Hit Detection →
Package Building →
DAQ Protocol Encoding →
Fast Serial Output Interface

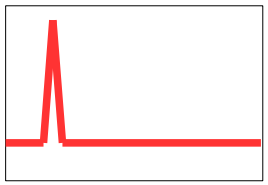
See next slides

Possible SPADIC user

- **TRD**
- Maybe RICH
- MUCH ???
- ...

The processing chain of SPADIC 1.0 – Step 1

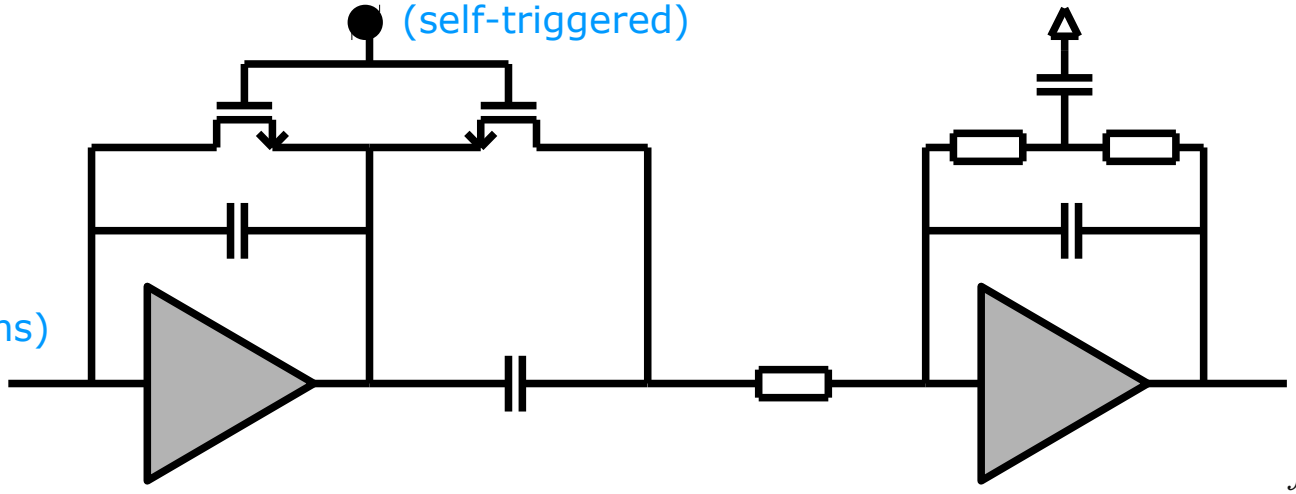
Step 1: Amplification + Shaping



Fast charge pulse from detector (< 10ns)

(RICH: negative, up to 2pF)

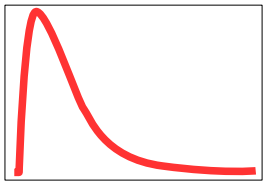
Continuous rest (self-triggered)



Charge Amplifier

2nd Order Shaper

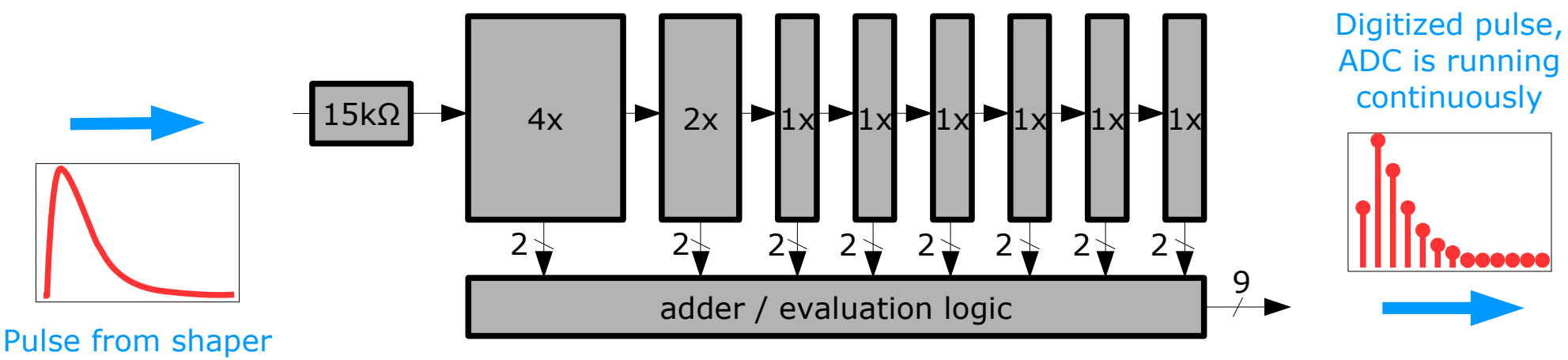
Voltage pulse, shape depends only on shaping-time:



$$f(t) = \left(\frac{t}{T}\right)^2 \exp\left(\frac{t}{T}\right)$$

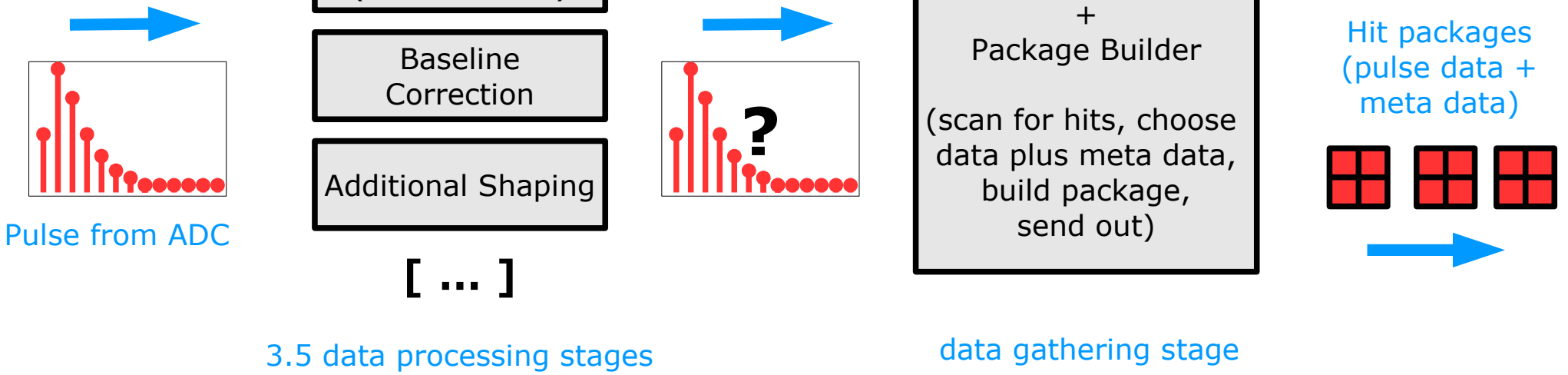
The processing chain of SPADIC 1.0 – Step 2

Step 2: Digitization



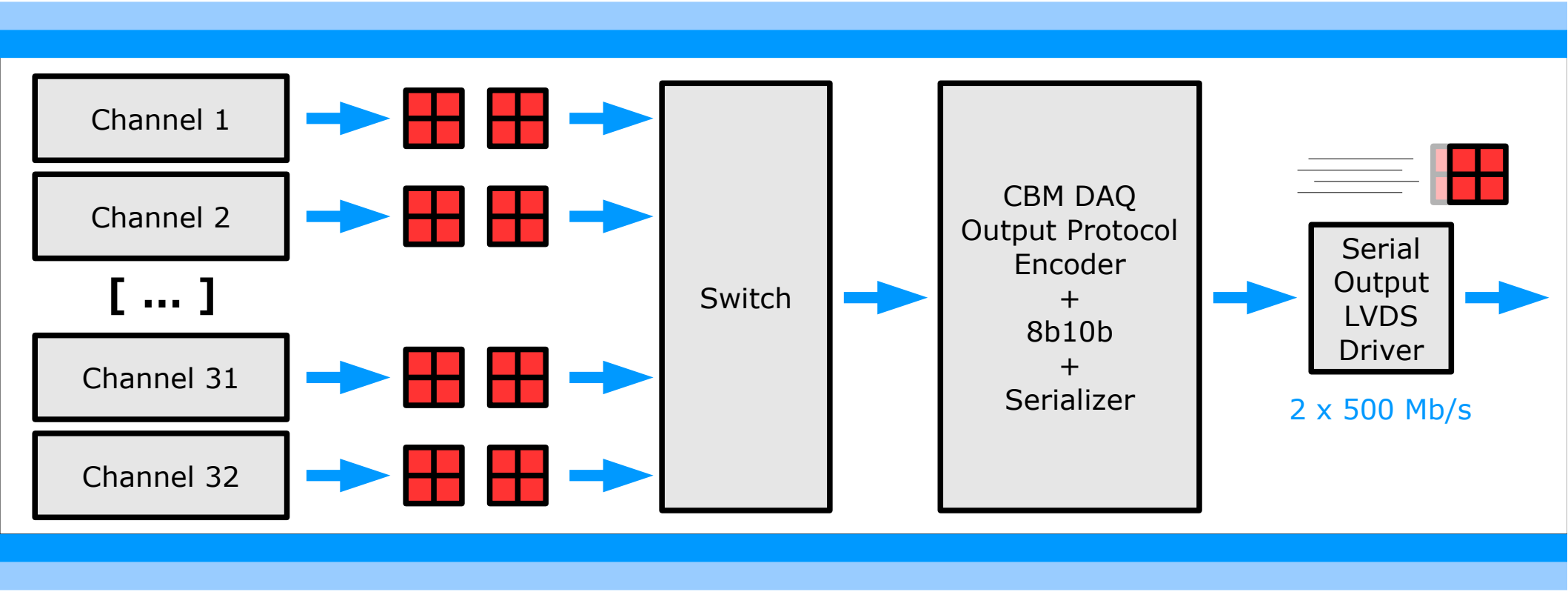
The processing chain of SPADIC 1.0 – Step 3

Step 3: Data processing + Data Gathering



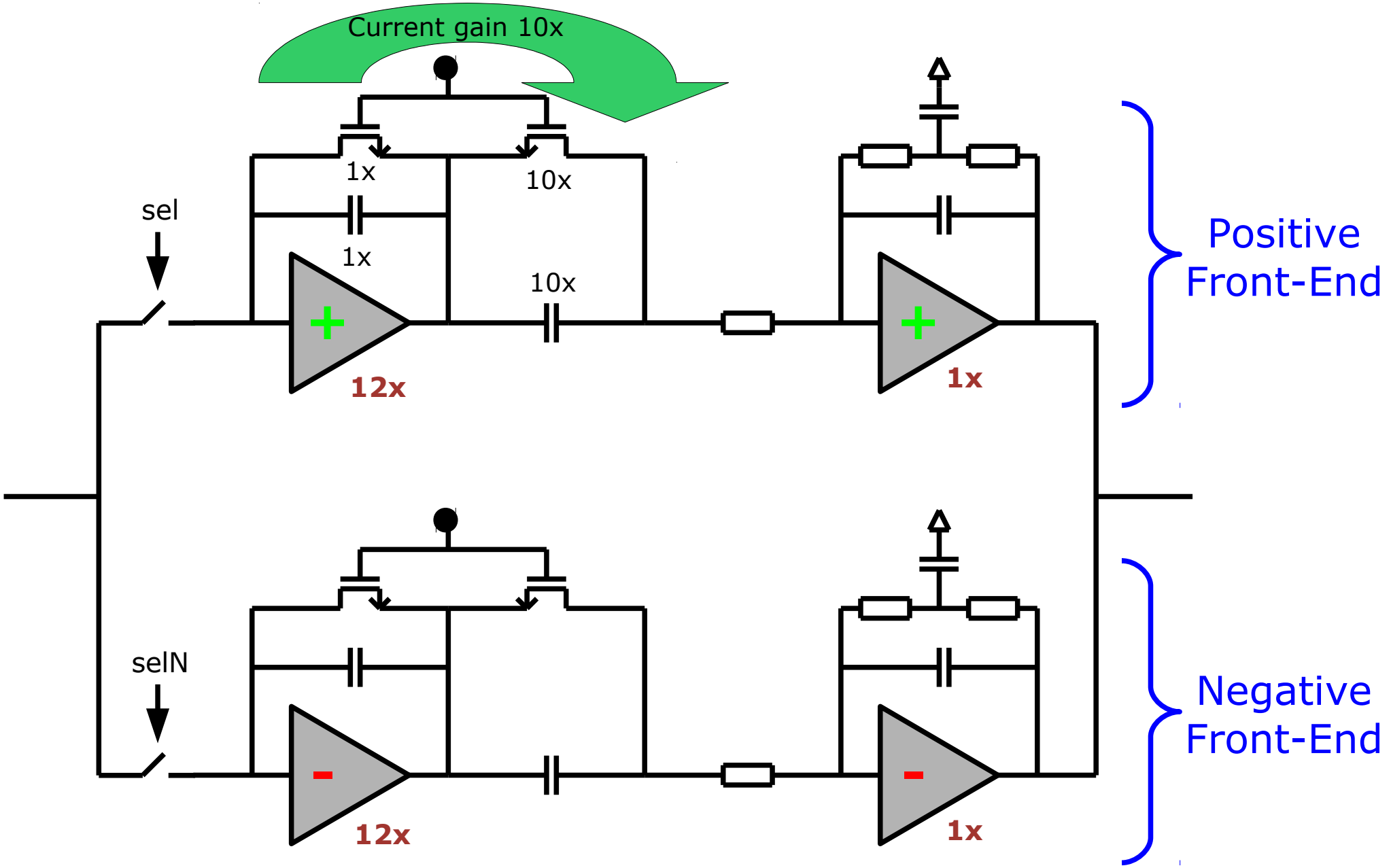
The processing chain of SPADIC 1.0 – Step 4

Step 4: Inter-Channel Network and Output Protocol

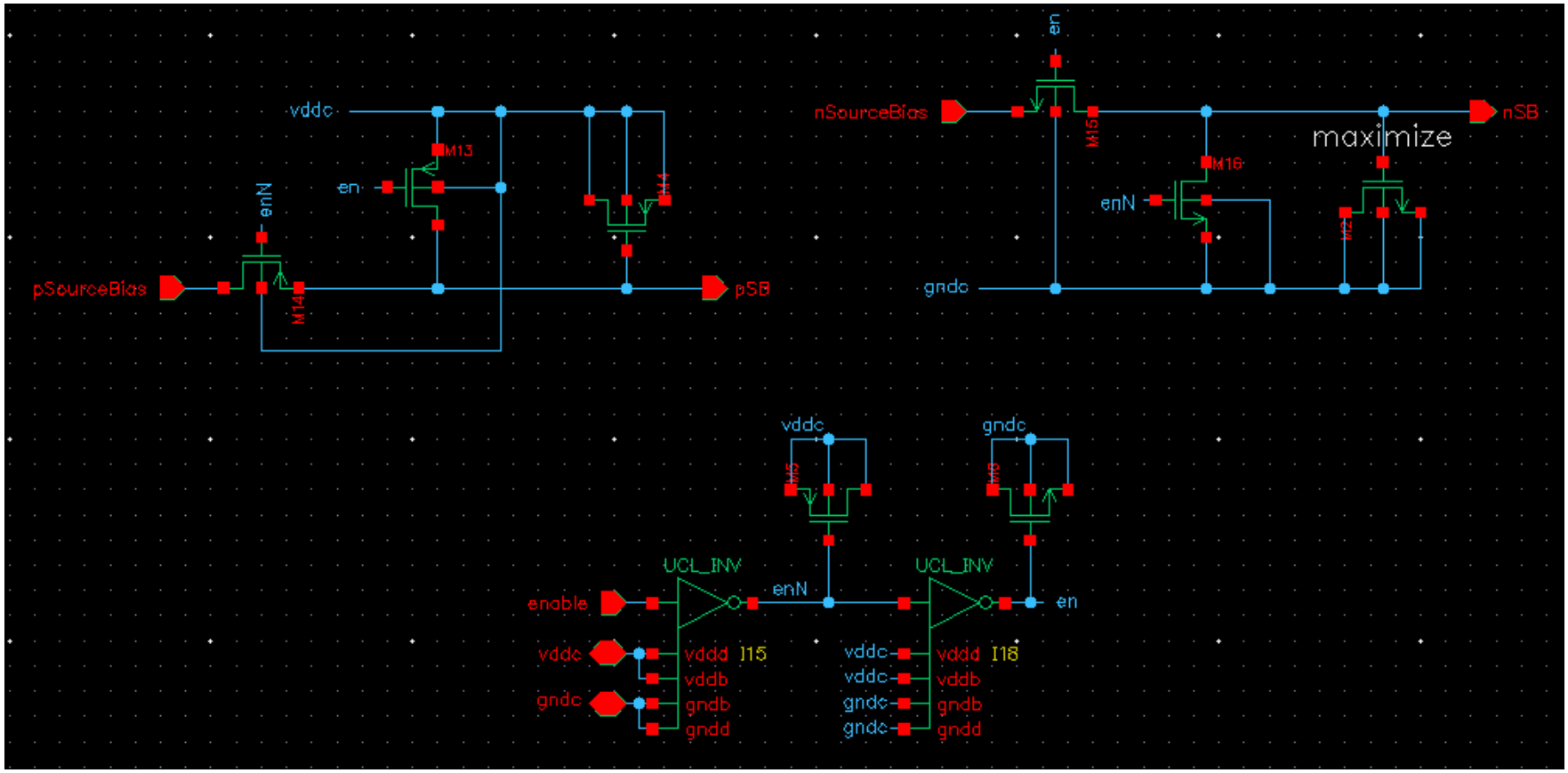


2. Selected Details of Analog Part

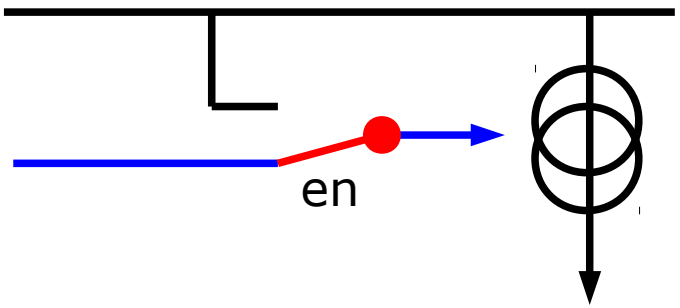
Two Front-End Amplifiers



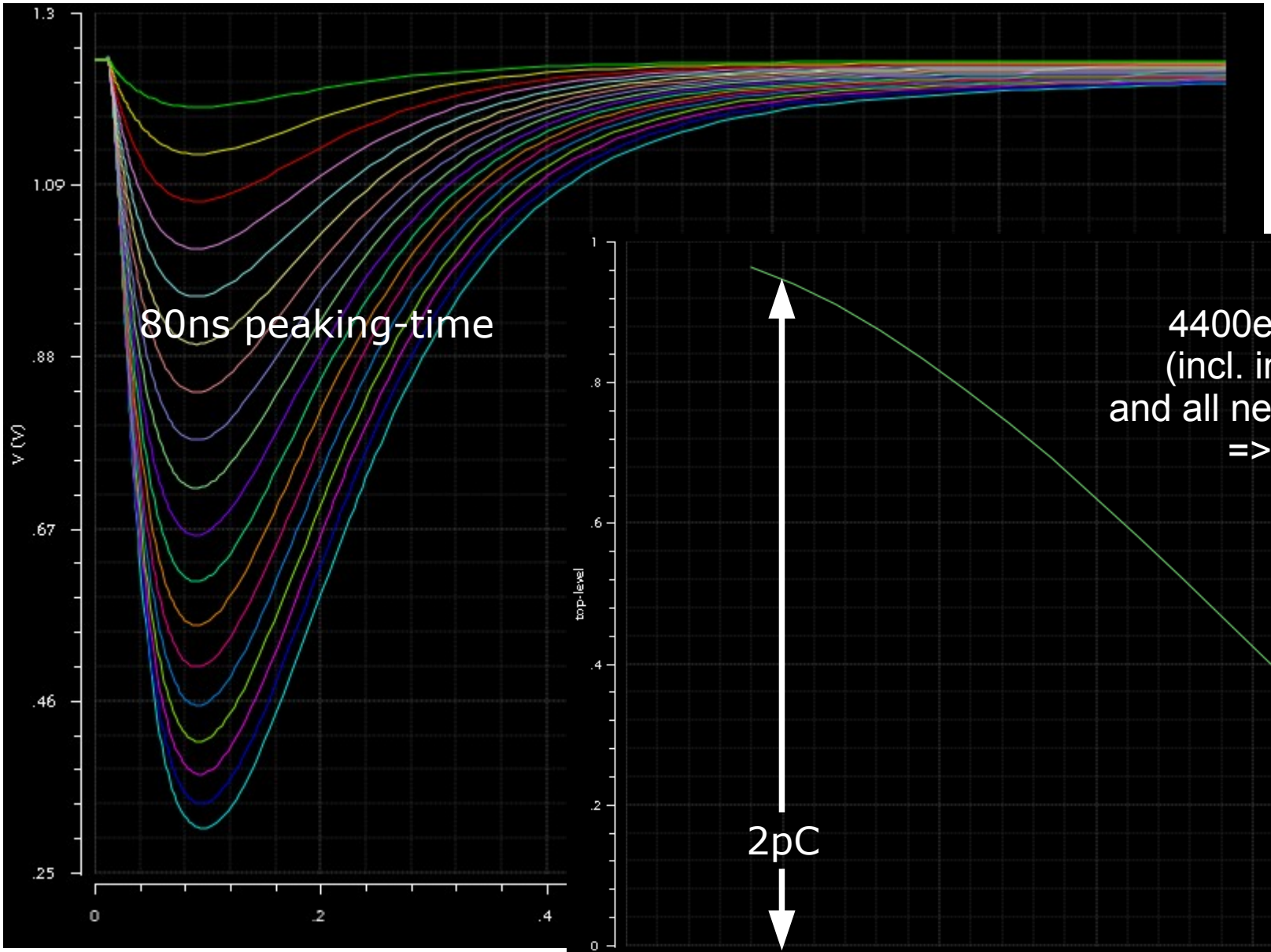
Front-End Enable



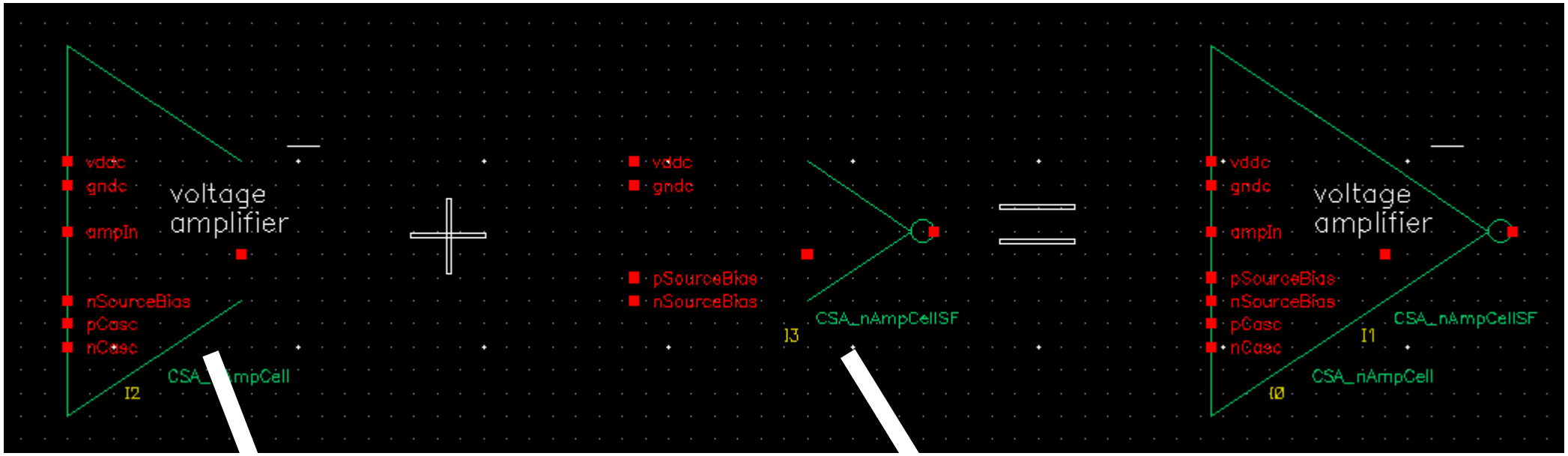
Trick: Connect gates of current sources to vdd/gnd if disabled => Noise!!

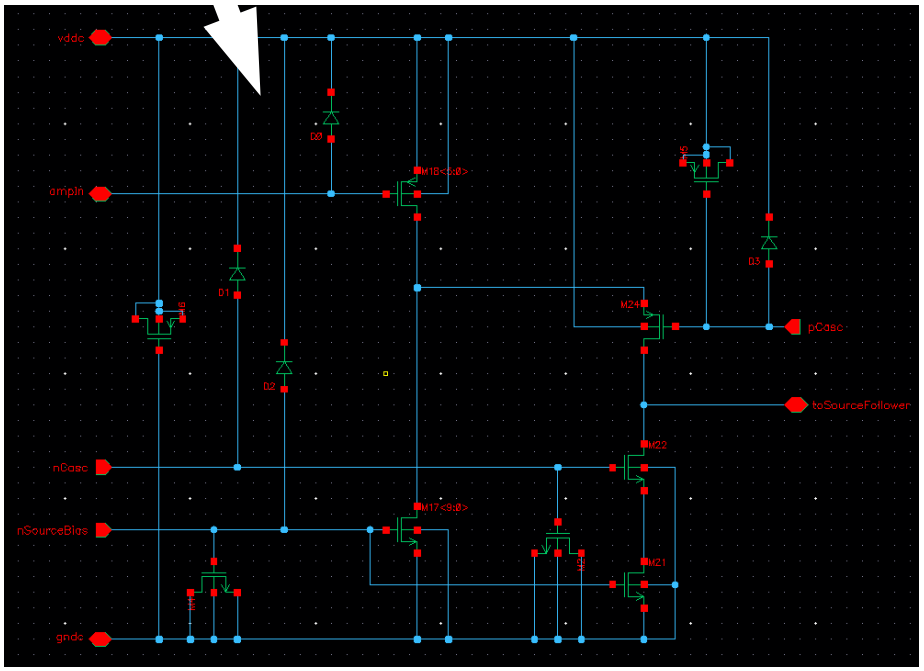


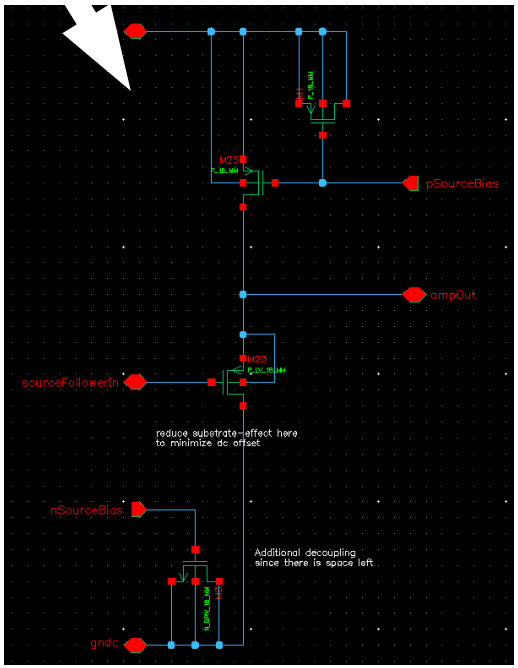
Negative Front-End (old → low gain)



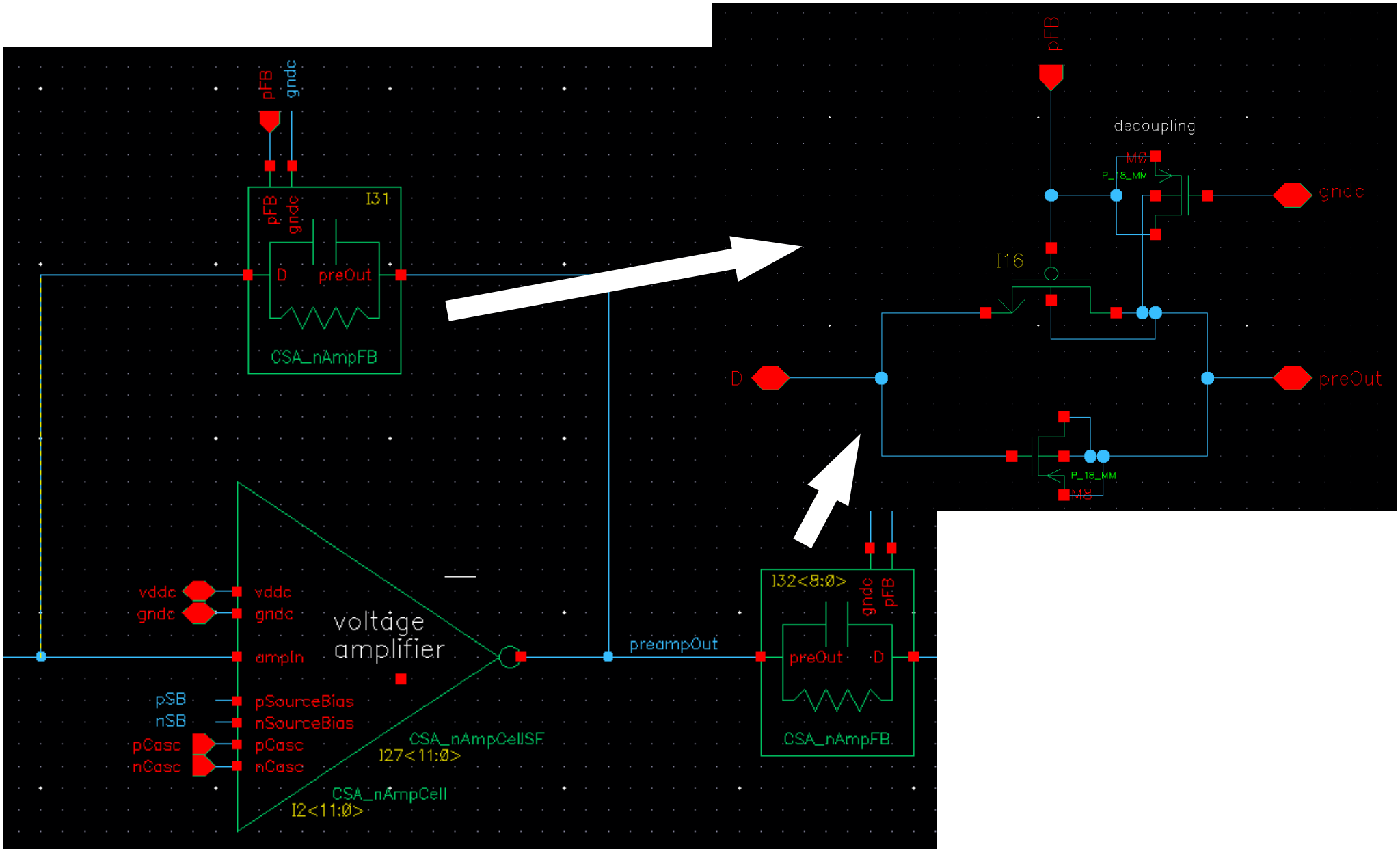
Modular Design – Example 1: Amplifier Cell



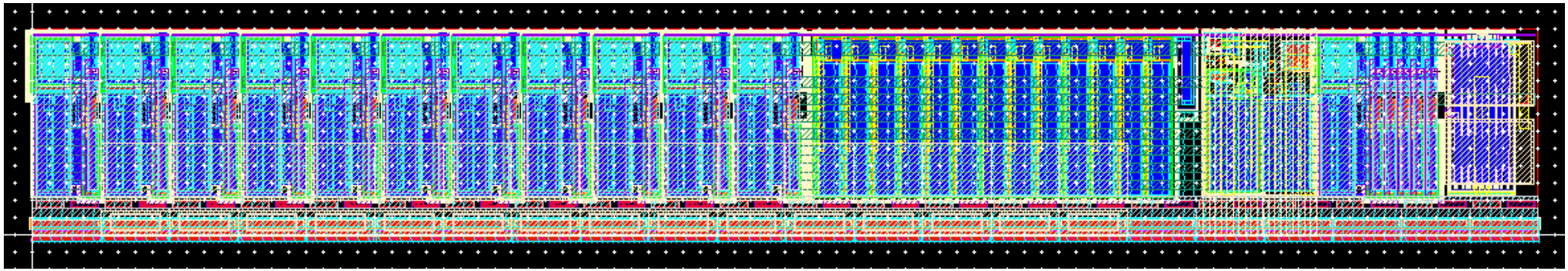




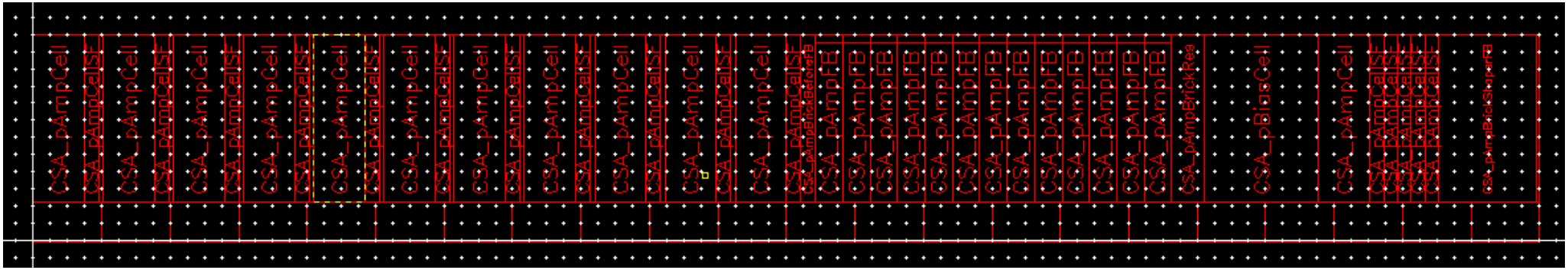
Modular Design – Example 2: Feedback



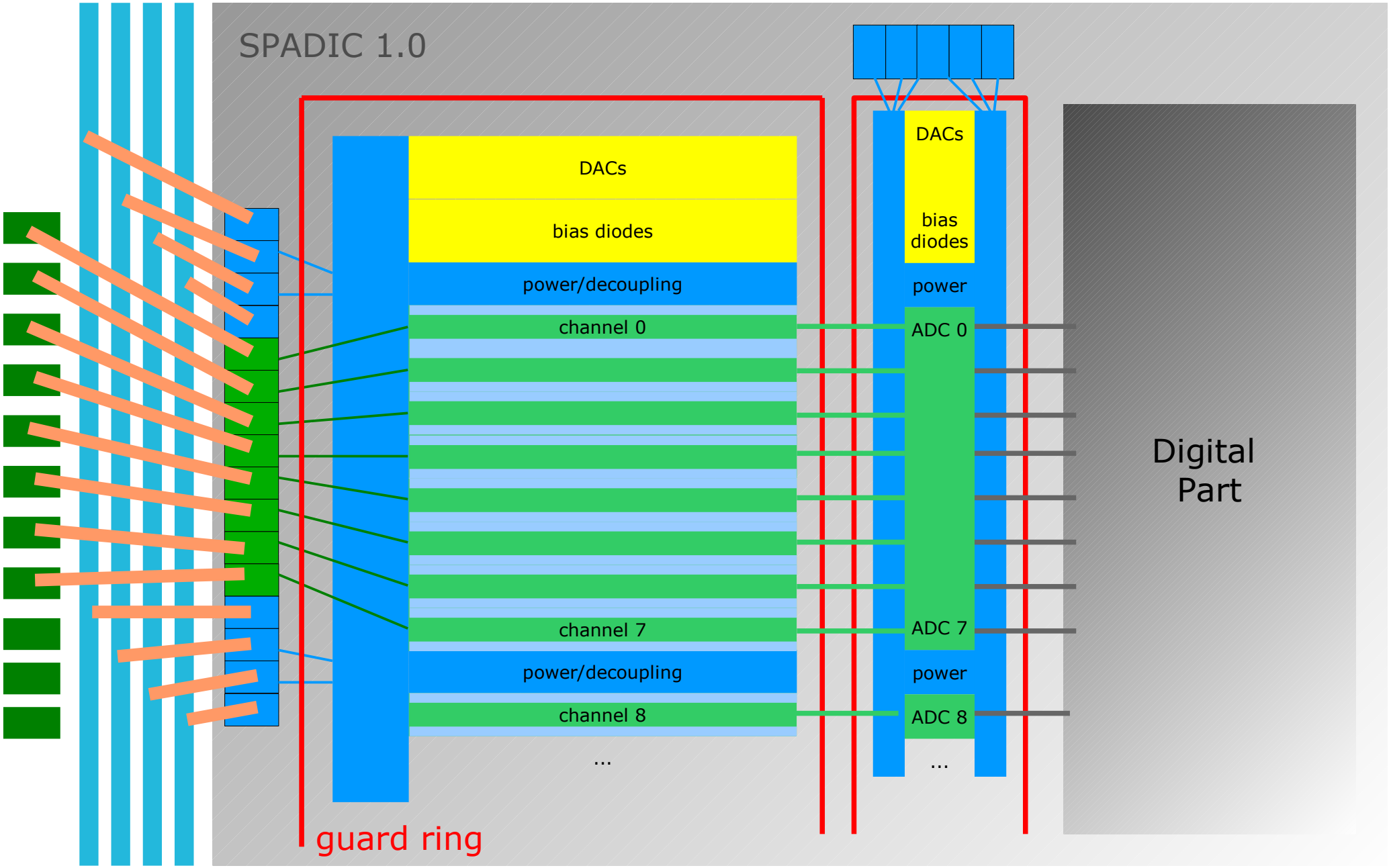
Modular Design – Example 3: Channel Bricks



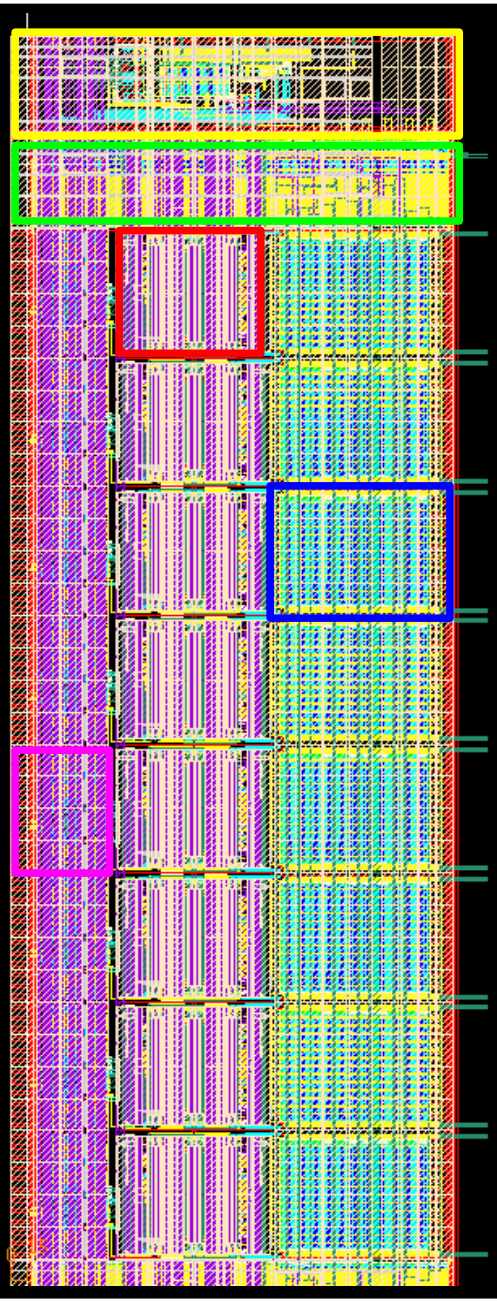
Modules wherever redundancy occurs
=> In all levels of hierarchy



CSA Power + Bias Scheme

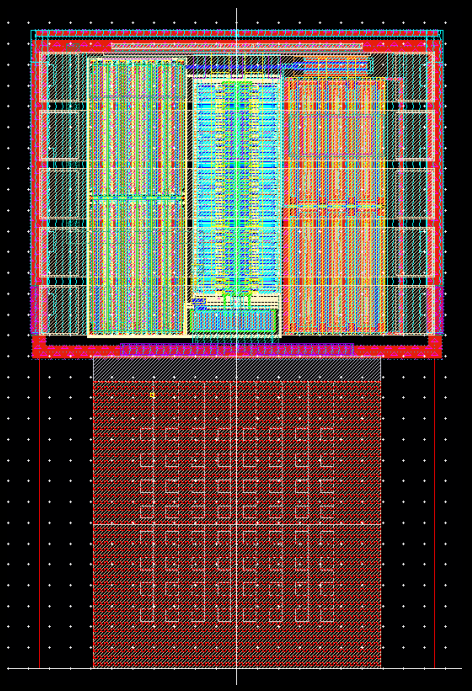


ADC Overview

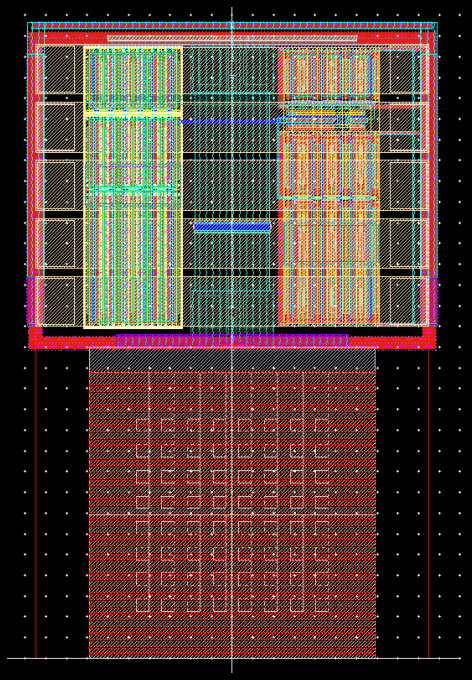


Pipeline ADC (32x)
Input Cell (32x)
Decoupling + Power Cell (32x)
Bias Diodes (1x)
Power channel + Decoupling (5x)

Pads



I2C



Protected

[...]

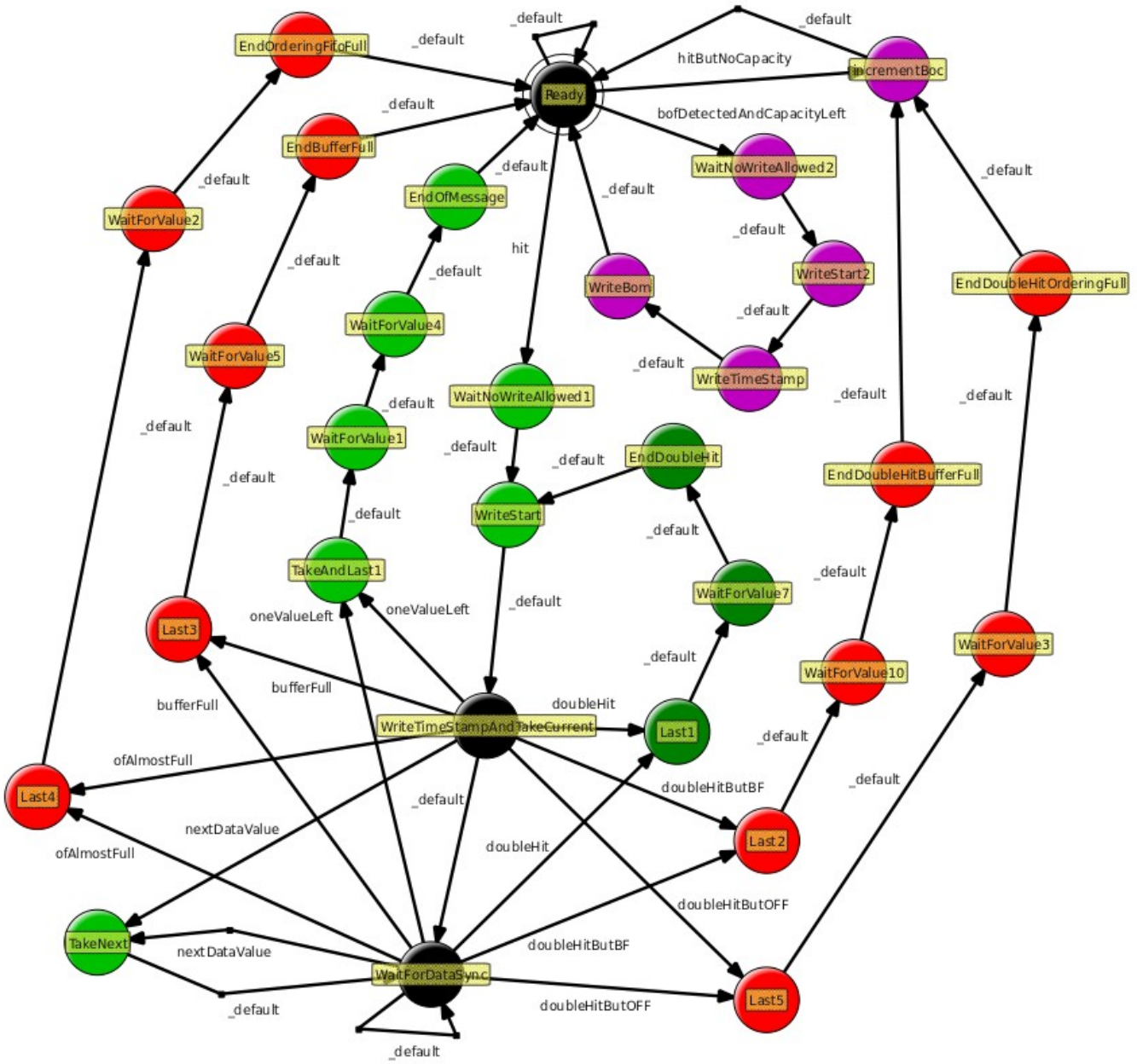
All SPADIC pads were copied from SUSLIB_PADS + updated
Added some new pads: I2C, Analog Input with serial res, Pull-Up, …
New Pitch: 95um (old was 80um)

3. Selected Details of Digital Part

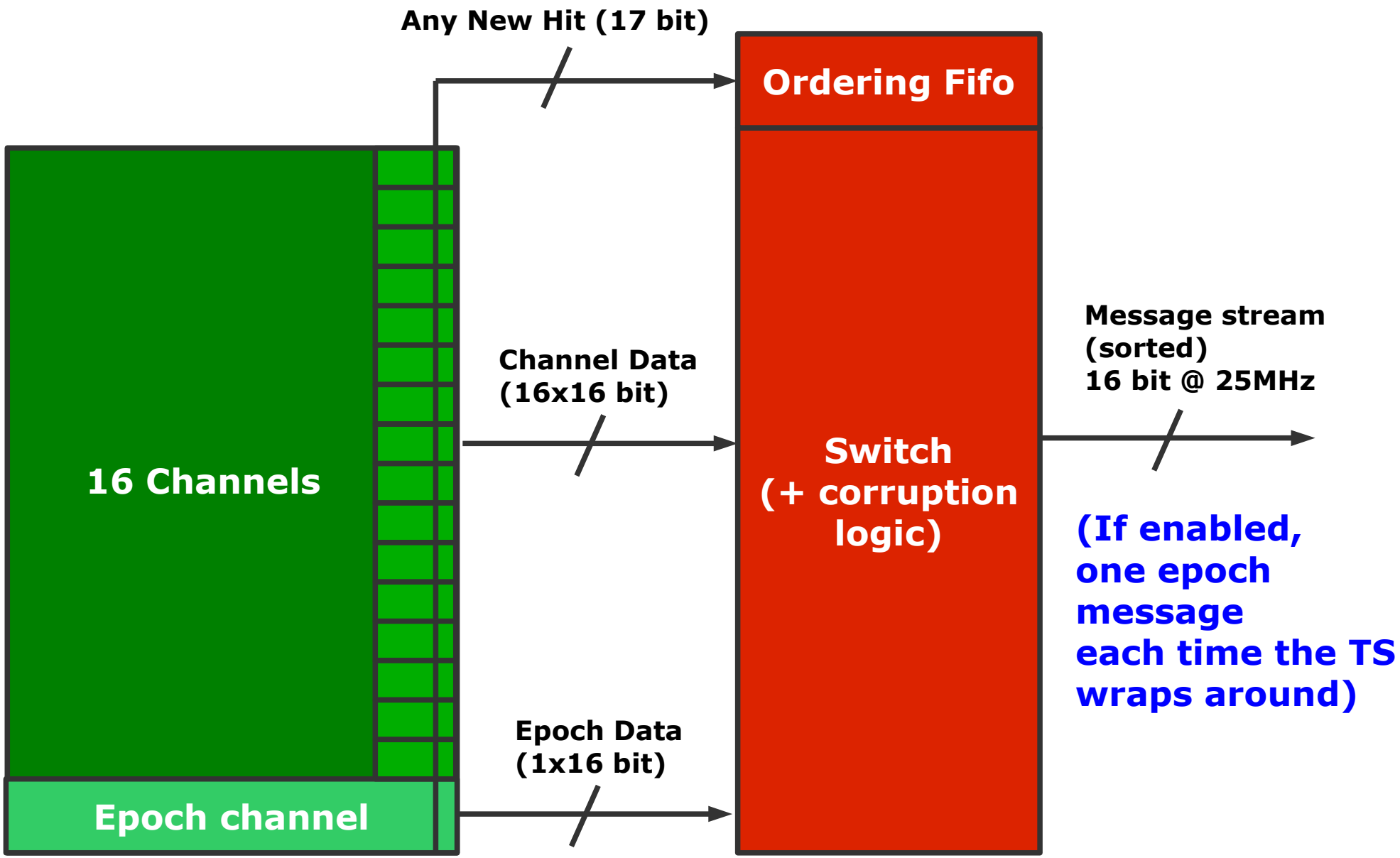
Hit Logic

See black board
(sorry, no time left)

Show:
Input Delay
Hit Detector
Hit Sync
Main FSM
Data Package Wrapper
Meta Data Generator
Output Fifo



Channel Switch + Epoch Channel



Message format

4.3 Example Messages

normal hit message (n-word message)		
1000 ssss ssss iiii	start of message	
1001 tttt tttt tttt	time-stamp	
1010 dddd dddd dddd	start of raw data	
0ddd dddd dddd dddd	continued raw data	
[...]	continued raw data	
0ddd dddd dddd dddd	continued raw data	
1011 pppp pphh -sss	end of message	
buffer overflow message (3-word message)		
1000 ssss ssss iiii	start of message	
1001 tttt tttt tttt	time-stamp	could be removed later in stream
1100 -- bbbb bbbb	buffer overflow counter	end of message
epoch marker message (2-word message)		
1000 ssss ssss 0000	start, channel-id fixed to 0 (necessary for arbitration)	can be removed later in stream
1101 eeee eeee eeee	epoch marker	end of message
extracted hit message (n-word message)		
1000 ssss ssss iiii	start of message	
1001 tttt tttt tttt	time-stamp	
1110 dddd dddd dddd	start of extracted data	
0ddd dddd dddd dddd	continued extracted data	
1011 pppp pphh -sss	end of message	
empty word info message (1 word only)		
1111 0101 -- --	emtpy word	can be removed anytime
empty words within normal hit message (n-word message)		
1000 ssss ssss iiii	start of message	
1111 0101 -- --	emtpy word	can be removed anytime
1001 tttt tttt tttt	time-stamp	
1111 0101 -- --	emtpy word	can be removed anytime
1010 dddd dddd dddd	start of raw data	
0ddd dddd dddd dddd	continued raw data	
[...]	continued raw data	
0ddd dddd dddd dddd	continued raw data	
1111 0101 -- --	emtpy word	can be removed anytime
1011 pppp pphh -sss	end of message	

Synchronization

CBMnet feature: DLM (deterministic latency messages)

From SPADIC's point of view:
DLM0: Loop back (used to measure delay through CBMnet)
DLM1: Sync signal (always @ TS wrap-around, set new TS + opt: epoch)
DLM2: With next DLM1 set TS
DLM8: Start readout
DLM9: Stop readout
[...]

Problem: Very few documentation available → basically one time-diagram

But: (Hopefully) good Verilog file for simulation (timing checks included)

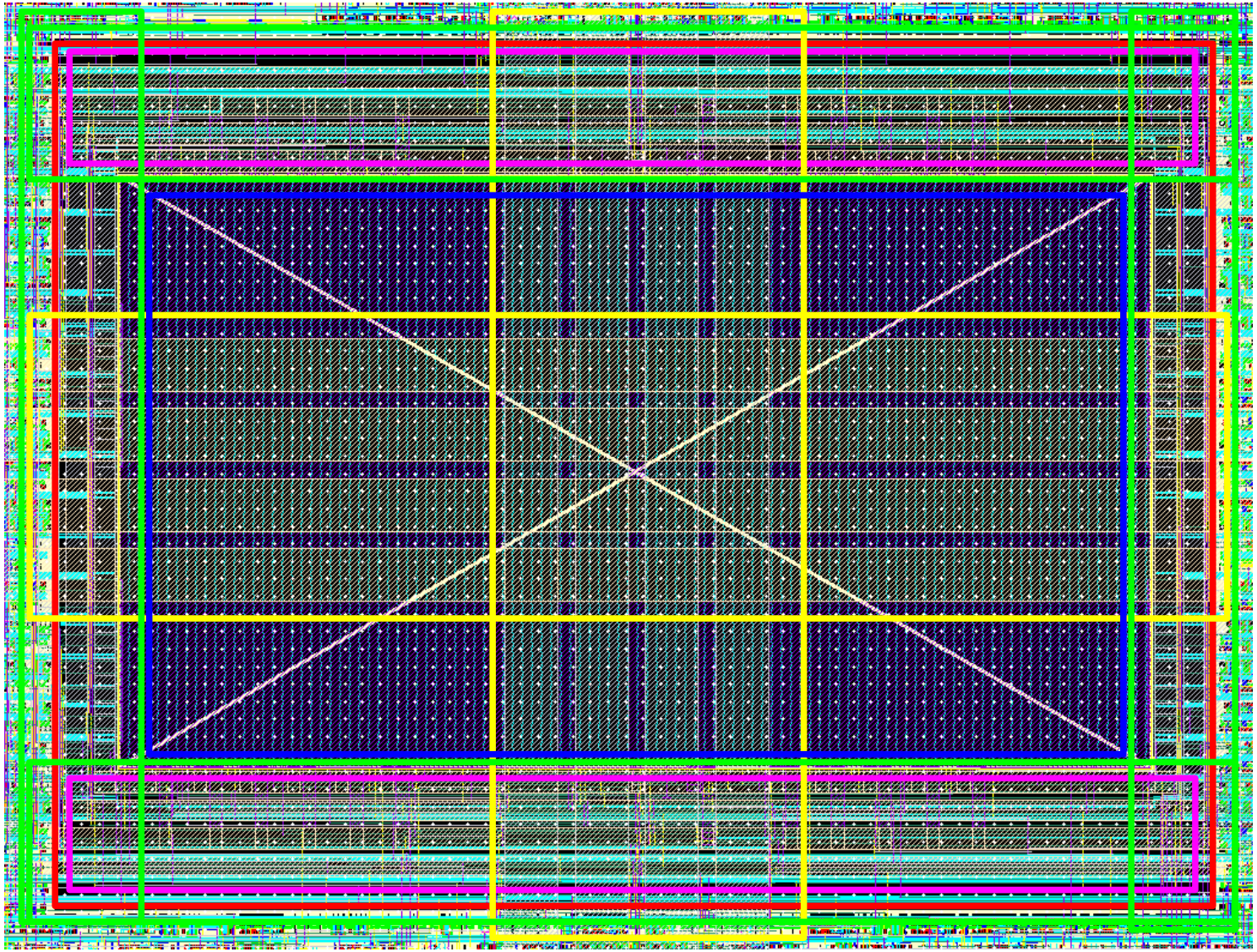
```
04    //ram signals
05  `ifdef SYN //for synthesis
06    assign AD = wptr;
07    assign BD = rptr;
08    assign DIA = dataIn;
09    assign DIB = {DATASIZE{1'b0}};
10    assign OEA = ~REG_disable;
11    assign OEB = ~REG_disable;
12    assign WEAN = ~shiftIn_clean;
13    assign WEBN = 1'b1;
14    assign CSA = ~REG_disable;
15    assign CSB = ~REG_disable;
16    assign dataOutRamB = DOB;
17  `else
18    assign #10 AD = wptr;
19    assign #10 BD = rptr;
20    assign #10 DIA = dataIn;
21    assign #10 DIB = {DATASIZE{1'b0}};
22    assign #10 OEA = ~REG_disable;
23    assign #10 OEB = ~REG_disable;
24    assign #10 WEAN = ~shiftIn_clean;
25    assign #10 WEBN = 1'b1;
26    assign #10 CSA = ~REG_disable;
27    assign #10 CSB = ~REG_disable;
28    assign dataOutRamB = DOB;
29  `endif
30
```

was 1'b0!

Very dangerous delay constructs for simulation necessary => indeed copy and paste bug

Faraday SRAMS, Part 2: Black-box Routing

Many problems in ENC run when using the SRAM-libfile without further constraining the floorplan



blockage from libfile

standard-cell cutout

routing blockage M1

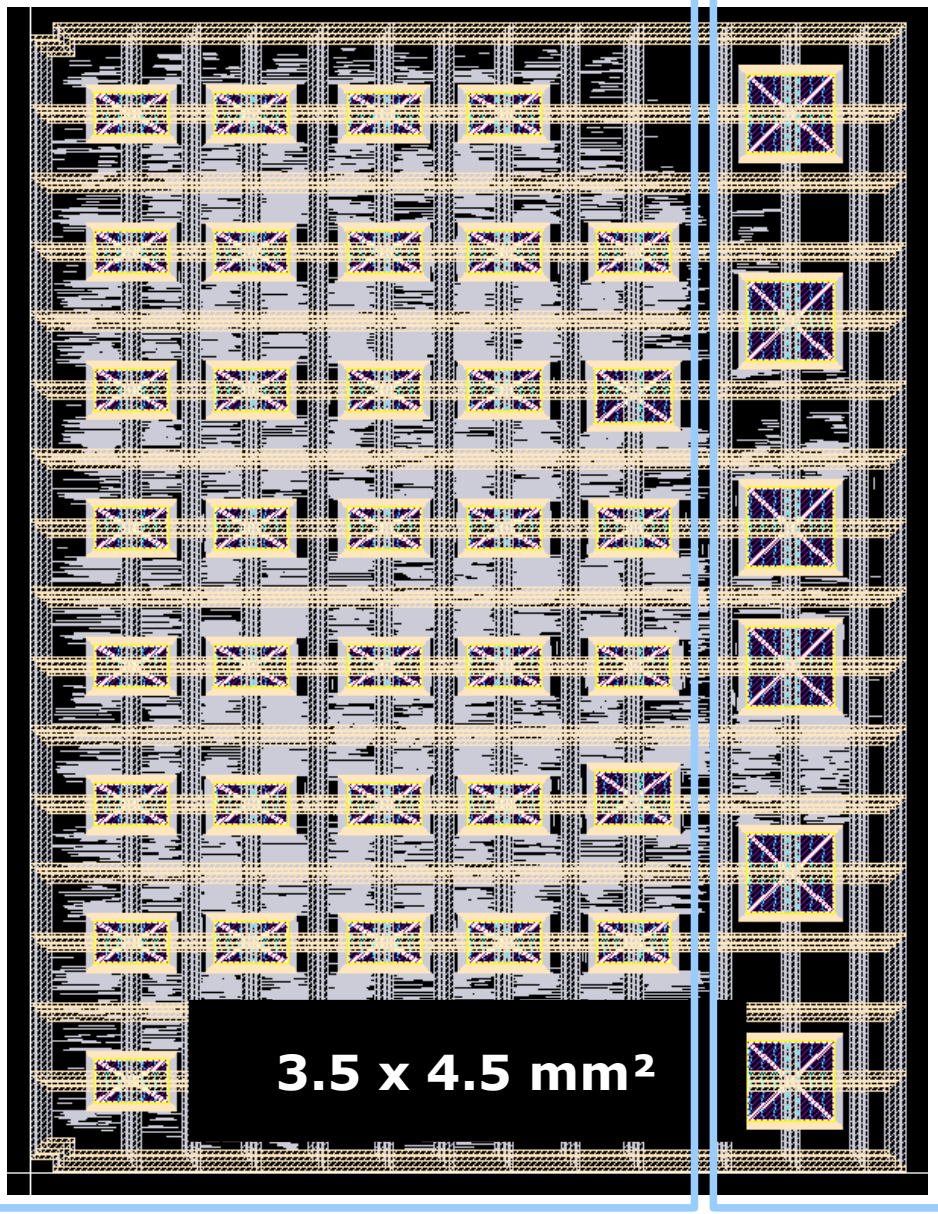
power stripes crossing just above srams

power ring (sroute did not connect sram properly by default)

Faraday SRAMS, Part 3: Floorplanning

Very, very important: Placement pattern of srams resp. floorplaning

SuS Logic
38 srams



RA Logic
(CBM net)
6 srams

Faraday SRAMS, Part 4: LEF Import

```
via12 = techFindViaDefByName(tech "M1_M2")
via23 = techFindViaDefByName(tech "M2_M3")
via34 = techFindViaDefByName(tech "M3_M4")
viaParams = list(list("originOffset" list(0.3 0.3)))
xylist=list()
labelCounter=0
textDisplayCounter=0
gndFound=0
vccFound=0
(foreach shape cv~>shapes
  (if  (eq shape->objType "textDisplay") then
    (if  not(exists( x xylist x == shape~>xy)) then
      ;get properties
      labeltext=shape~>associate~>name
      xypos=shape~>xy
```

Problem: Even for black-box LVS at least dummy cell for SRAMS required => Cadence LEF Import very corrupt: Strange symbols on non-existent layers, unconnected pins with same label, ... => Skill script for repairing the imported layout

Still no LVS from complete design possible → with digital part as BB only

```
            "lowerLeft" "R0" "stick" 0.3
            )
            gndFound=1
          )
        else
          (if (labeltext == "VCC") then
            (if (vccFound == 0) then
              ;createLabel
              (dbCreateLabel
               cv
               '("M4_CAD" "TEXT")
               xypos
               labeltext
               "lowerLeft" "R0" "stick" 0.3
               )
```

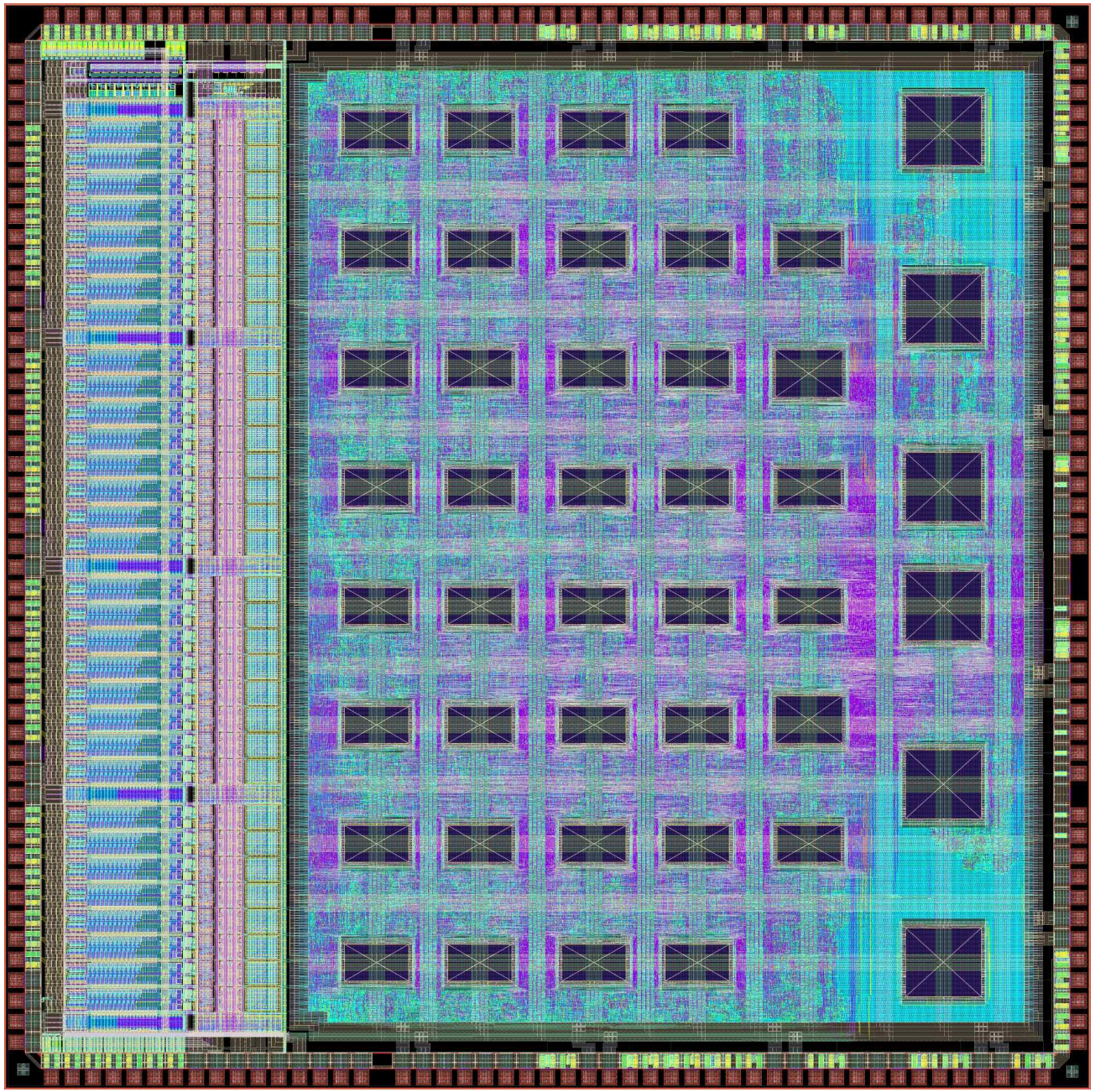
Synthesis/ENC Summary

Few words to a long story:

- **Modified synthesis/encounter run scripts based on Jochen's, Florian's and Richard's sources => should sometime merge all**
- **Analog part finished for weeks, only digital part still under construction**
- **Most difficult parts (for a beginner): Floor-planning + Constraints – the rest is more or trial and error**
- **A complete run (syn+enc) takes about 8h**
- **About 30-60% of the runs abort due to segv (resp. memory corruptions, already detected during run), other reasons were e.g: NFS read-only (for a couple of minutes), no license left, home directory full, timeout when encounter tries to start parallel threads …)**
- **=> Very stressful rhythm: fix a small thing, start at least 3 runs in parallel, wait for 8h, check all results, fix a small thing, …**
- **Verilog adjustments till the last minute (and beyond)**
- **First Post-Place and route simulation one day after submission date**
- **Bad, bad problems with first PPNR sim due to annotation problems**
- **Finally: Submission on 6.11. (delayed by 4 days)**

4. Summary

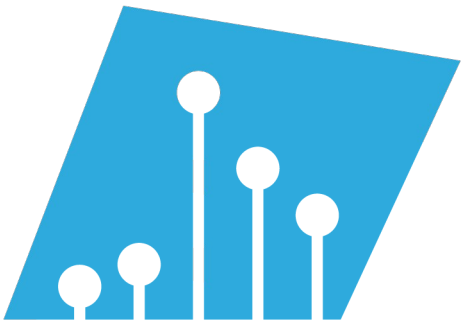
Top-Level Layout



2 SPADIC 1.0 Documentation

2.1 General

Parameter	Design-Value	Meas-Value	Comment
Technology	UMC 180 nm		
Number of metal layers	6		
Number of channels	32		
Channel pitch	120 µm		
Pad pitch	95 µm		
Chip size	4.96 mm × 4.96 mm		3 × 3 Mini@sics
Total power consumption			
	Analog Part	Digital Part	Total
Number of transistors	226 k	2.09 M	2.32 M
Number of nets	58.1 k	1.02 M	1.08 M
Number of pads			191
Number of sram blocks		44	44
Total sram memory		4.4 kByte	4.4 kByte
Number of DACs	48		48
Number of config registers	576	1270	1846
Total wire length		14.39 m	
Area	1.12 mm × 4.65 mm	3.46 mm × 4.54 mm	4.96 mm × 4.96 mm



SPADIC

Self triggered Pulse Amplification and Digitization asIC

http://www.spadic.uni-hd.de